Document Ref: ZYBS-BDS
Issue No: 1. July 1982

Greenbank Electronics
Telephone: 051-645 3391

Interak 1

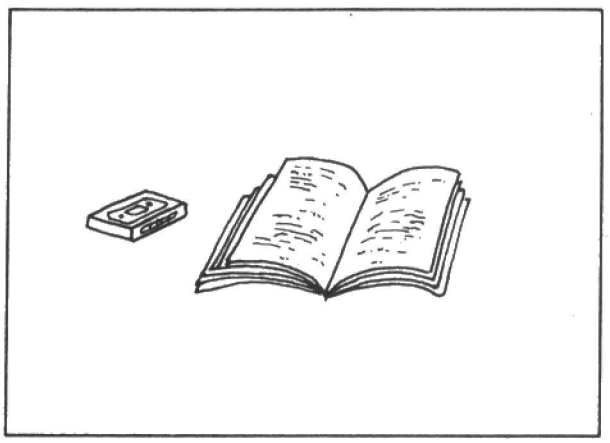
ZYBASIC BASIC

Floating Point BASIC for Interak 1 System

Floating Point BASIC for Interak 1 System

FEATURES

* DTI-1 Cassette Card Compatible.
* Loads at 2400 Baud.
* Floating-point arithmetic from $\pm 1.5 \times 10^{-39}$ to $\pm 1.7 \times 10^{+38}$.
* Hexadecimal input option.
* 260 Floating Point Variables + 26 String Variables (each of max. length 255 characters.)
* Runs in RAM at 4 MHz with no Wait States. (EPROM based version available).
* Will work without full 48K Interak 1 RAM, needs only one extra 16K RAM card over minimum system.
* Enhanced Command Set, for Use with Interak 1 VDU-K. (Will also work with Kemitron VDU-A,B,G with reduced performance.)
* Certain Graphics Commands. (Pixel SET, RESET, POINT).
* Low Cost.
* Uses no Memory Space when not Loaded, Unlike ROM-Based Interpreters (leaves space clear for e.g. machine code Chess Program.)



* Includes Printer Driver for Hard-copy Output if RS-232 Interface is Available.
* Printer on/off Control. (If printer is available.)
* Supplied with Temporary Manual, (Comprehensive Manual paying particular attention to beginner's needs is in preparation).
* Manual may be purchased in advance.

DESCRIPTION

This is a version of the Palo Alto BASIC, specially expanded to include some special commands which take advantage of the VDU-K pixel graphic characters. Also the syntax has been carefully revised so that it is more in line with the popular "commercial" computers. Some special commands (DON, DOFF, LINE, SCROLL, incremental POKE, a more useful variation than DOKE for multiple POKE commands) have been introduced. These are particularly useful in enabling a pleasing screen output to be achieved (for example to permit instructions for games to be printed without scrolling up the games "board" itself). Various subtle refinements have been incorporated, which would only be appreciated if they had been left out, e.g. the cursor is left on the screen only when an input is expected, this makes for a more pleasing display since the presence of an unwanted cursor is most unprofessional.

It is recommended that beginners start by learning BASIC and writing simple programs using this language, before they progress to machine code/assembly language programming. (This is a similar recommendation as saying you should learn to drive an automatic motor car before you go on to one with a manual gear-change. In computing as with motoring others may of course hold different opinions.)

6K is quite small for a BASIC by today's standards, but this 6K is nevertheless quite powerful. Remember 6K of Z80A code can do quite a bit more than 6K of code for some less powerful microprocessor.

Even when much larger BASICs come along a beginner is still recommended to begin with this one. This size of BASIC is just about the right size to be "friendly" enough on which to learn. The main obstacle to learning is fear of the power of the machine, and big BASICs are very frightening at first because of their necessary complexity.

Simple BASIC programs are very easy to understand without formal tuition, e.g.

```
10 INPUT A
20 INPUT B
30 PRINT "The sum is", A+B
40 PRINT "The difference is", A-B
50 GOTO 10
```

or,

```
10 . . . . . .
20 . . . . . .
30 PRINT "Tell me your name"
40 N$ = INPUT$
50 PRINT "Hello", N$
60 . . . . . .
```

Playing about with simple programs like this, i.e ones which can easily be understood and indeed written by a beginner, will enable the essence of computing to be absorbed in the most practical way - by doing.

Although simply using someone else's program is not so clever, there are a few sample programs included in the Manual, plotting rectangles, circles and the like, and a few simple games such as Dice Pontoon, Hangman, Noughts and Crosses and so on, donated by members of the informal User's Group which started with ZYMON 1 some time ago. The outstanding game is one called "Monster Mash", in which the computer develops a random maze and populates it with four monsters, which have to be destroyed by the user. Some people have described this game as being of Amusement Arcade quality, and it certainly represents an encouragement to the beginner to see what can be achieved when the Interak 1 system is properly programmed.

RAM-BASED

The fact that the BASIC runs in RAM and therefore uses no memory when it is not loaded is an obvious but by no means trivial point. The whole point of a computer, in fact almost the definition of a computer, is that it will do different things on different days, according to how it is programmed. When working with large machine-code programs it is a positive hindrance to have such items as large BASIC interpreters cluttering up the memory space.

It is certainly not inconceivable that an Interak 1 user might be wanting to use his system to write his own BASIC, perhaps for amusement, or perhaps to sell to other users. Such a person will of course want to develop and debug his program in RAM at its proper address, and he certainly would have a more difficult task if there was already a BASIC "squatting" there in ROM, in permanent residence right where he wanted to be.

If a BASIC is in ROM (or EPROM), the cost of the memory devices themselves will have been entirely wasted should a user eventually upgrade to disks, since once disks are present the normal professional practice is to store all programs on disk, with a nice clean RAM area to receive the different programs when they are run. (It is acknowledged that such systems as Tangerine, Apple, TRS-80, PET, etc. retain their ROMs when disks are added, but this is preferable to the firms concerned than facing the embarrassment of having to tell their customers to throw away all the ROMs they paid for when they bought the system!)

Loading from cassette does of course take time, but at the chosen baud rate of 2400 the waiting time is minimised. At the traditional speed of 300 baud the wait would have been so long that the technique could not really have been considered practical.

It is certainly hoped that no "bugs" are present in this BASIC, but if any were found it would be a simple matter to load an updated tape, without needing to dismantle the computer in any way. Contrast this with the ZX81, which has been sold to many people with a "bug" in its arithmetic routines, (i.e. it gets its sums wrong!) The only "fix" is to scrap the ROM which contains the BASIC, and send off for a new one, which many users are reluctant to do. In the highly unlikely (we hope!) event of a "bug" being found in the ZYBASIC then the "fix" would be simple, and there would be nothing to throw away.

Despite all of the arguments put forward above an alternative version can be supplied at extra cost in 4 x 2716 EPROMs, to satisfy anyone who has strong feelings against the use of RAM. Four EPROMs are necessary as the program is a little over 6K in length, and so the last EPROM is mainly empty.

ORDERING INFORMATION

ZYBASIC is normally supplied on a tape cassette for loading into RAM at hex. address A000; this version should be ordered as ZYBASIC 2A. For those users who prefer it, it can also be supplied in 4 EPROMs (2K 5V type), which are to be located at hex. address C000; this version should be ordered as ZYBASIC 2C. The Manual is priced separately so that it may be purchased in advance if required; remember therefore if you do not already have it to include the Manual on your order when purchasing the ZYBASIC.

SPECIFICATION

A brief specification of the commands, functions, etc. available is given on the next page. Many of the various items will be recognisable from their names, but reference to the Manual will be required for some of the more unusual and unique items. The need for this fuller explanation is one of the reasons that the Manual has been made available for separate purchase.

SPECIFICATION

Variables

A-Z, A1-Z1, ... , A9-Z9. (260 variables in all).

A$-Z$. 26 string variables, each of up to 255 characters in length (Needs full Interak 1 RAM).

Keyboard Control

CTRL-C	Return to Command Mode.
CTRL-L	Clear Screen.
CTRL-N	Get next (editor).
CTRL-R	Get Rest (editor).
CTRL-S	Output Stall (Any Key Continues)

Commands

@ (int)
? (=PRINT)
! (=REM)
AUTO
BYE
CALL
CLS
DATA
DON
DOFF
ED numb
FOR expr TO expr STEP NEXT
GOTO int
GOSUB int RETURN
IF
LET
LET string
LEN(string)
MID$(string,expr,expr)
LINE
LIST int, int
NEW
OLD
ON var GOTO int
ON var GOSUB int,int
OUT port,data,data (inc)
PAGE
POKE addr,data,data (inc)
PRINT
PRON

Commands (Continued)

PROFF
RND
READ var
REM
RESET (int,int)
RESTORE
RUN
SAVE
SCROLL
SET (int,int)
STOP
TAB (int)

Abbreviations

The commands do not have to be typed in full in a program; the Manual gives details on their abbreviated forms.

Functions

ABS (expr)
COS (expr)
COSR (expr)
FREE
IN var
INT (expr)
INKEY var
INPUT var
POINT (int,int)
PEEK (int)
SIN (expr)
SINR (expr)
SQR (expr)
USR

Operators

( ) Parentheses.
+,- Unary plus and minus.
*,/ Multiplication and division.
+,- Addition and subtraction.
>=,<>,>,=,<=,< Relational Operators.
(var=expr)+(var>expr) Arithmetic OR.
(var<expr)*(var=expr) Arithmetic AND.
=,<> Relation of strings.
+ Concatenation of strings.
"" String delimiters.